Center for Satellite and Hybrid Communication Networks

Automated Monitoring and Management of Hybrid Broadband Networks

Faculty:	J.Baras, N. Roussopoulos
Research Staff:	G. Mykoniatis
Graduate Students:	A. Arora, V. Birmani, A. Gupta, H. Li, A. Pinho, H. Xi
Industry Support:	Hughes Network Systems, Lockheed Martin Global Telecommunications, Bellcore
Industry Interest:	Bell Atlantic, IBM, Motorola, Hewlett-Packard, Teledesic-Motorola, AT&T
Other Sponsors:	ARL ATIRP Consortium

Industry Advisory Board Meeting
February 17, 1999

Commercial Objectives and Significance

- **Objectives:**
  - Develop intelligent monitoring systems for satellite and hybrid network fault and performance management
  - Develop schemes for efficient storage and aggregation of massive network monitoring data
- **Significance:**
  - Management of hybrid networks is a critical market differentiator and critical for network existence and operation
  - Heterogeneity and increasing speed call for management automation and “non-conventional” ideas

Support for NASA Missions: Objectives and Significance

- **Objectives:**
  - Develop network management systems for NASA networks and space communication systems using commercial satellites
  - Develop autonomous intelligent network management and control systems for complex spacecraft
- **Significance:**
  - As NASA moves towards use of commercial satellite networks and the Internet intelligent fault and performance management becomes vital

CSHCN

Delay Monitoring

- **ATM Networks give QoS guarantees to connections**
- **QoS means bounds on**
  - Maximum Cell transfer Delay (CTD)
  - Cell Delay Variance (CDV)
  - Cell Loss Ratio (CLR)
- **Measurement and control of QoS being provided is most important for real time connections**
- **We concentrate on delay monitoring**
  - Optional timestamp field permits the measurement of round trip time
  - CTD from source to destination cannot be measured
  - Frequency of the measurements is not optimal: OAM cells are inserted after a fixed number of cells (128 or more)

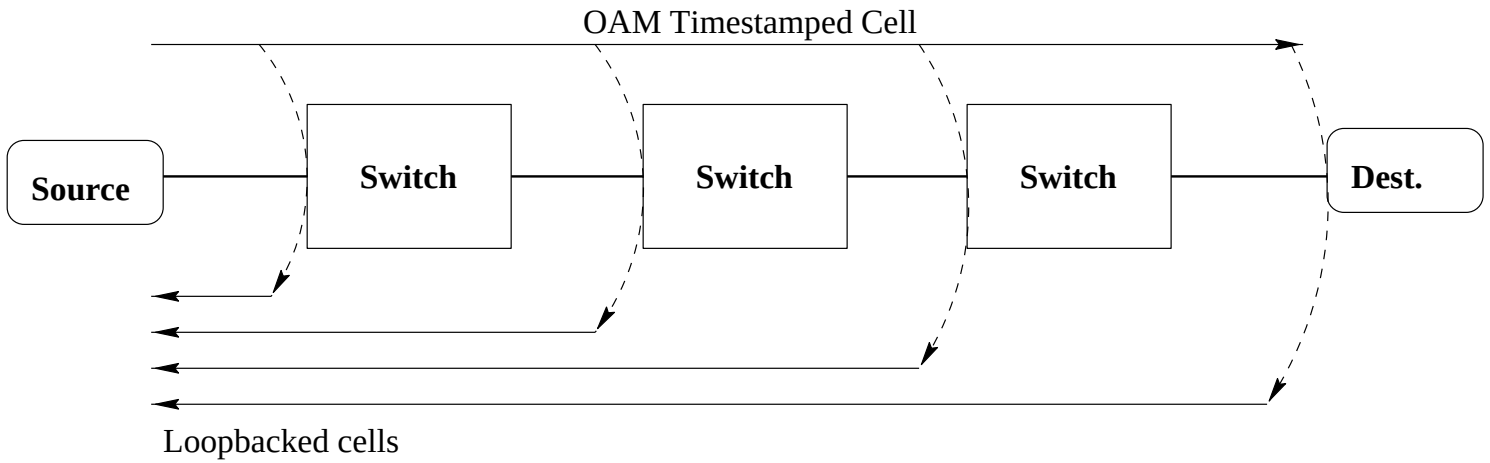
CSHCN

Measurement Cells

- Cell delay: measured by breaking it into delays experienced at each switch
- Queuing delay: measured by time- stamping cell at ingress and reading at egress of a switch
- Each switch writes delay of the cell on it
- Cons: difficult to implement, requires new processing capabilities at switches

Loopback Based Measurement

- Back-flooding: OAM cell is looped back by all switches in the path
- Gives an estimate of the incremental round trip time to each switch
- Very high overhead though easily implementable as loopback OAM cells can be used



- Selective back-flooding: Loopback performed only if queuing delay at the node is above a threshold. This gives lower bandwidth overhead

Sampling Frequency

- **The rate of inserting measurement cells should be minimum without loosing any important trend information**
- **Desired sampling algorithm**
  - start at a high rate
  - reduce rate when pattern is identified
  - increase the rate when a shift in the pattern is detected and decrease as soon as possible
  - keep redefining the stable rate

Current Algorithm

- **Algorithm:**
  - Convert the last N delay samples to an analog signal using LPF of bandwidth B
  - Find the Nyquist sampling frequency, F_{samp}
  - Use min(F_{samp},F_{max}) to calculate the time of insertion of the next sample
  - Advance the window
- **Future research includes a better formulation of the algorithm and investigation of trade-off between accuracy and overhead**

Efficient Computation of Aggregates of Network Management Data in Large Satellite Networks

- **Main Objectives**
  - Design efficient methods for computing, storing and manipulating the spatial and temporal aggregates of monitored network statistical data
  - Develop flexible tools for defining and using hierarchies
  - Compute aggregates across time, attributes, sessions, users, calls, beams, satellites
  - Aggregates derived from “raw data” collected from the network every fifteen minutes or faster, for thousands of sessions per min
  - Validate Use of Oracle 8 Time Series Cartridges to store and maintain the spatial and temporal aggregates

Efficient Computation of Aggregates of Network Management Data

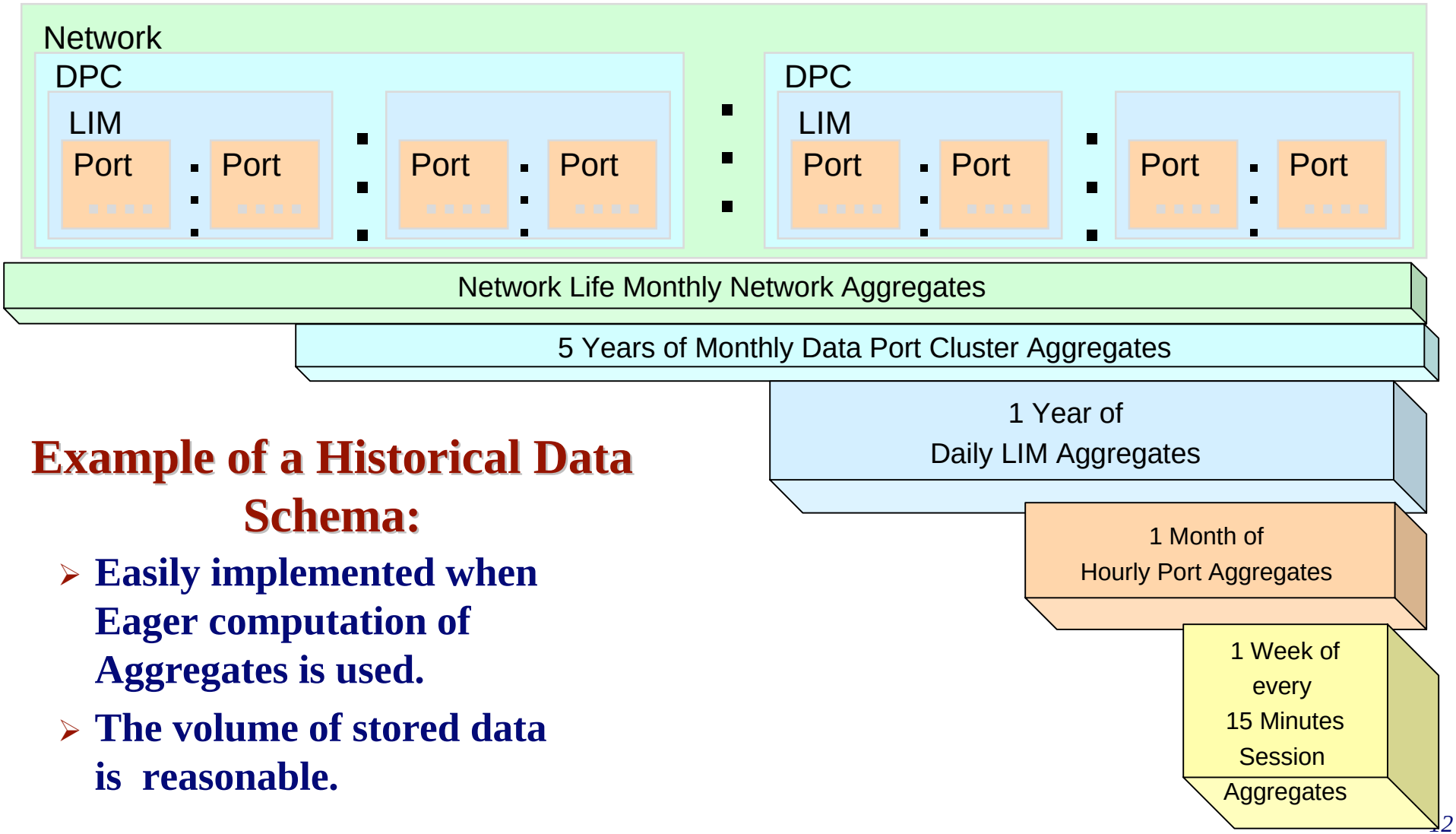
- **The network aggregations database system:**
  - converts “raw data” to delta-values and rates and loads them into the database
  - computes and stores spatial aggregates into the database
  - computes and stores temporal aggregates with different levels of granularity, providing finer granularity for recent data, and coarser granularity for older data, following an aging scheme
  - provides SQL-based interface for querying the stored aggregates
  - using appropriate SQL tools, supports arbitrary queries (e.g. list top ranked networks, list overloaded network elements or sessions, etc.), required by network monitoring
  - allows fast and frequent re-computation for very large data sets

Aggregates computation strategy: Eager *vs* Lazy

- On average query times when materialized aggregates are not available are 8 times longer.
- The more often queries are run to retrieve aggregated data, the more advantageous Eager aggregates computation becomes over Lazy aggregation.
- Eager aggregates computation allows the implementation of aging schemas to store historical data, with adequate granularity for each need, without wasting disk space.

CSHCN

A Schema



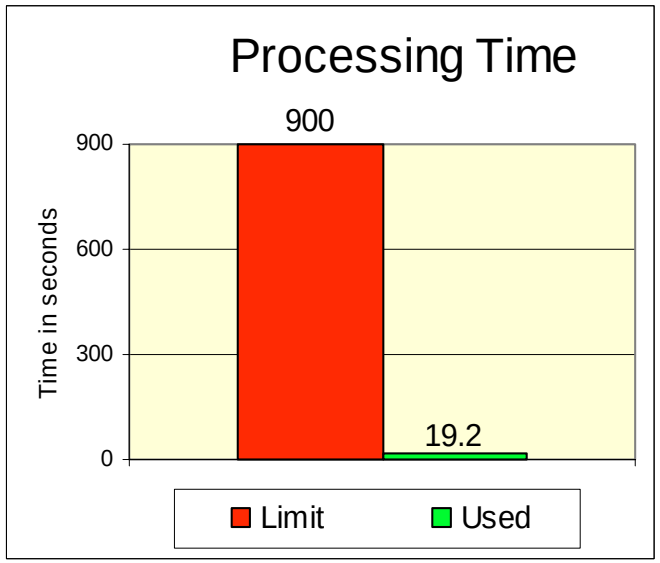
Example of a Historical Data Schema:

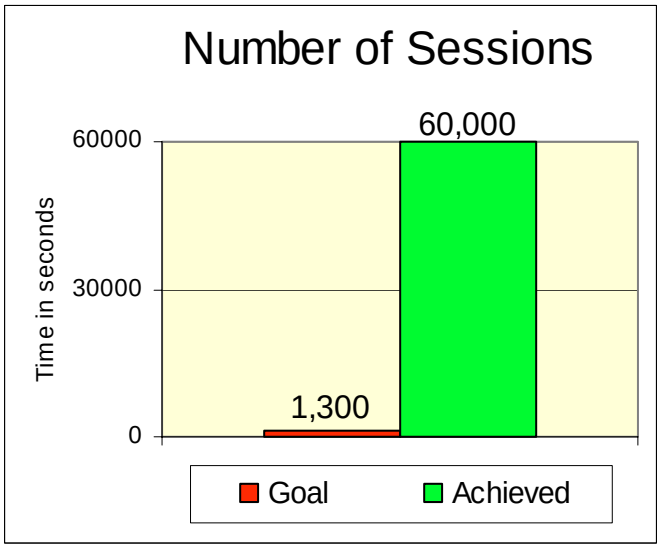
- Easily implemented when Eager computation of Aggregates is used.
- The volume of stored data is reasonable.

CSHCN

Main Goals *vs* Achievements:

- **Goals:**
  - Computation of aggregates of 1,300 sessions' statistics, each of them with 20 statistical attributes of network elements, every 15 minutes and store them on Oracle8 database, while running queries simulating the load generated by the operators.
- **Achievements:**
  - Computation of the aggregates of the 20 attributes by 11 dimensions for the 1,300 sessions and store them on the database in 19.2 seconds.
  - Computation of the aggregates of the 20 attributes by 11 dimensions for 60,000 sessions and store them on the database every 15 minutes (no queries running in parallel).





Adaptive Distributed Network Monitoring

- Large satellite constellations with millions of broadband users
  - Different user requirements and profiles
- As network size increases operators will become more-and-more incapable of digesting the information generated within the network
- Need for intelligence within the software of network control centers will become acute in the very near future
- Goal is to develop an adaptive distributed network monitoring system for retrieving, storing, accessing and processing statistics that are located in the network element management agents

Traditional Network Monitoring

- **Traditional Network Monitoring approach:**
  - centralized
  - based on polling

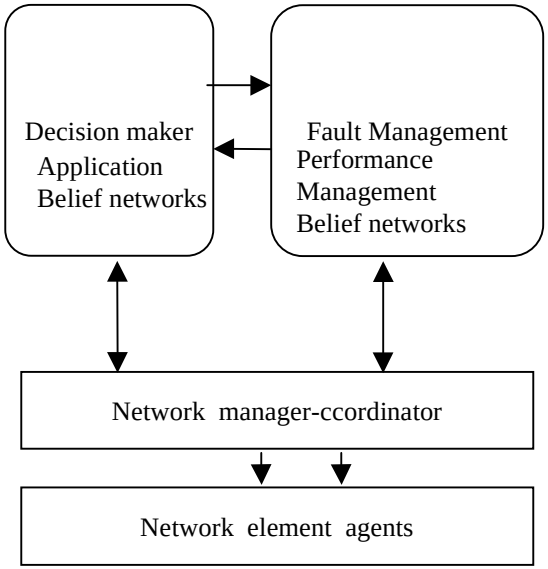
- **In each observation period:**
  - the **manager** retrieves the relevant information from all the network element management **agents** by polling
  - the manager processes the retrieved information (applies operations, computes aggregates, compares with threshold values, etc.)
  - based on the derived results and the specified policy the manager has to realize the supported network management functions (e.g. fault management, performance management, etc.)

Traditional Network Monitoring Drawbacks

- Huge amounts of data is travelling throughout the network to support monitoring
- A small amount of that data is really required by the manager to realize the supported network management functions
- Agents:
  - are hardwired applications,
  - have no ability to dynamically change the way the information is collected and aggregated
- The manager has no ability to change the way the information is reported
- Conventional monitoring methods will not suffice due to the scale, diversity and high performance requirements of broadband satellite constellations

Next Generation Network Monitoring

- **Three Components to provide flexibility, adaptability and intelligence**
  - Schemes that distribute and send to network elements Java programs that can:
    - re-direct data collection of elements
    - change the logic of the processing within elements
    - direct the elements to execute tests or collect new types of data
  - Belief networks to learn how to diagnose network faults quickly and accurately
  - Based on decisions by belief networks direct Java programs to most appropriate network elements for improved diagnosis and monitoring



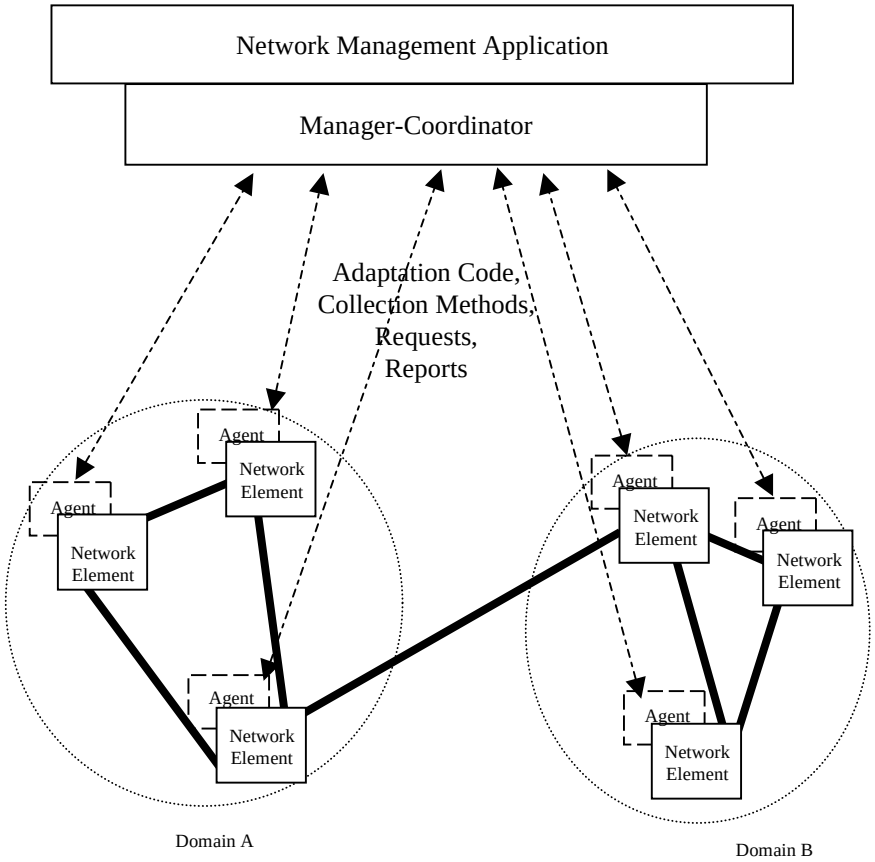
Dynamic Adaptive Distributed Network Monitoring

- **The defined architecture provides the ability to:**
  - dynamically define the view of the statistics offered by the network element agents, using filtering and spatial/temporal aggregation
  - deploy and maintain this view in the network elements
  - dynamically define trigger-conditions (e.g. threshold crossing)
  - provide reports when those conditions occur to the manager
  - dynamically change the policy used to report the results to the manager

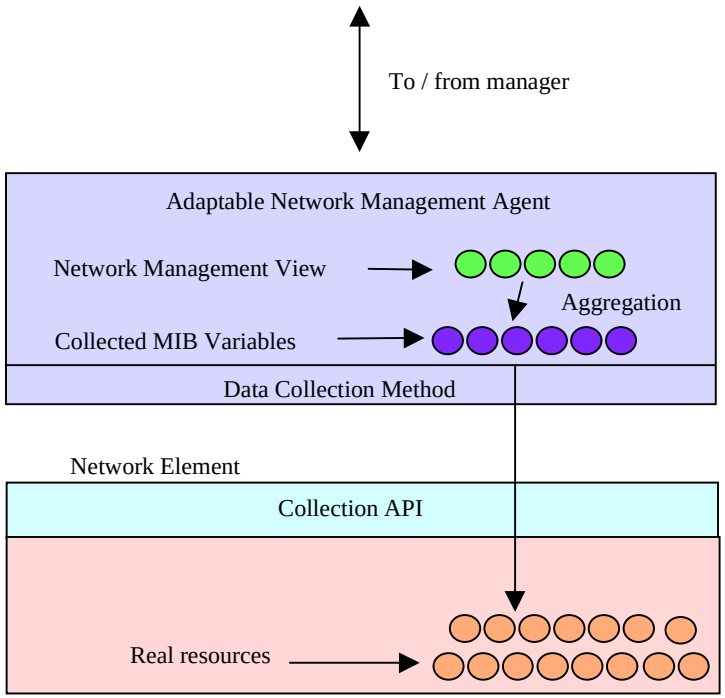
Dynamic Adaptable Distributed Network Monitoring Implementation

- The defined Dynamic Adaptable Distributed Architecture will be realized by:
  - the **Network Manager-Coordinator,**
  - a set of **Adaptable Network Element Management Agents**
  - network elements, supporting a **collection API**
- The functionality is “pushed” to the adaptable network element management agents, i.e. as close to the network elements as possible
- The monitoring information overhead to the network is minimized
- The “freshness” of the monitored information is maximized
- We will use a Java-based Mobile Agent technology

Components of Network Monitoring System

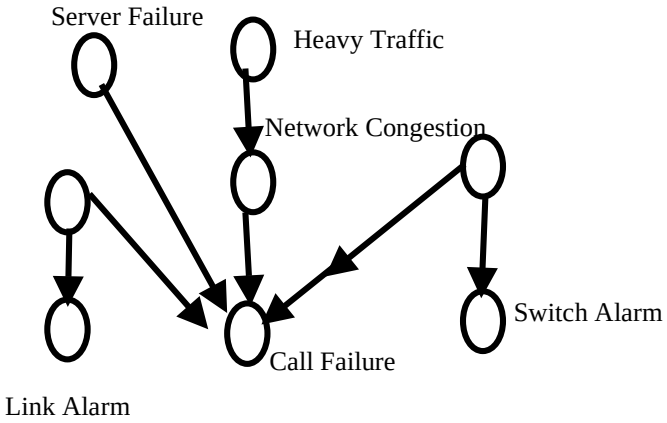


Network Element and Adaptable Network Element Management Agent



Belief Networks and Decision Directed Monitoring

- Intelligence embedded in the network manager-coordinator
- Will use belief networks for implementing intelligence and associated learning
- Will use reinforcement learning algorithms for implementing decision making
- Belief networks provide effective means to model causes and effects, and adaptable model for storing knowledge



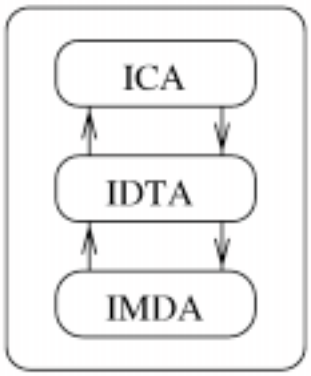
Configuration and Fault Management

- **Automated generation of OO network models, based on OMT, for configuration management**
- **Q-learning and reinforcement learning algorithms for dynamic reconfiguration**
- **Use of intelligent agents and active probing schemes to improve configuration management**
  - Send programs to network elements to compute statistics and report
- **Integrated methodology for configuration management and fault management**
  - Effects of configuration on fault identification
  - Configuration for best restoration
  - Configuration for maximum survivability
  - “Graphical grammar” for fault management visualization
  - Interactive graphics for statistics visualization and monitoring

Distributed, Intelligent Agent Based Fault Management Architecture

- **Developed architecture for fault management system based on a set of distributed asynchronously communicating agents**
  - Faults should be dealt with locally if they are local; only those that cannot be handled locally should draw global attention
  - Each intelligent agent of our proposed system uses belief networks to interpret data into beliefs, updating beliefs and in making inferences regarding the status of network elements
- **In the proposed architecture the managed network is divided into several domains and for each domain, there is an intelligent agent attached to it, which is responsible for this domain's fault management**
  - Domain may be a subnet, a cluster, a host or a member of a functional partition.
- **Domain Diagnostic Agent (DDA), with key functionalities: Fault Detection and Classification, Fault Localization and Identification, and Fault Correction**

Distributed, Intelligent Agent Based Fault Management Architecture



Components of a DDA

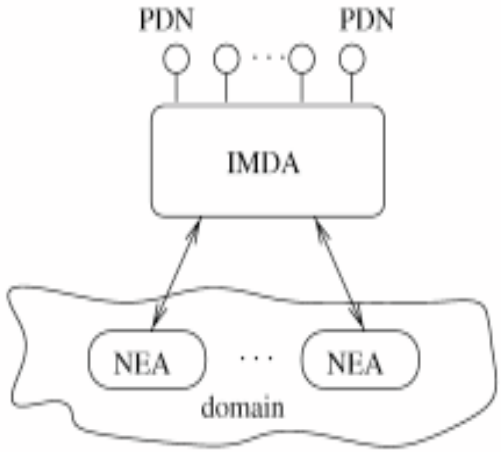


Illustration of an IMDA

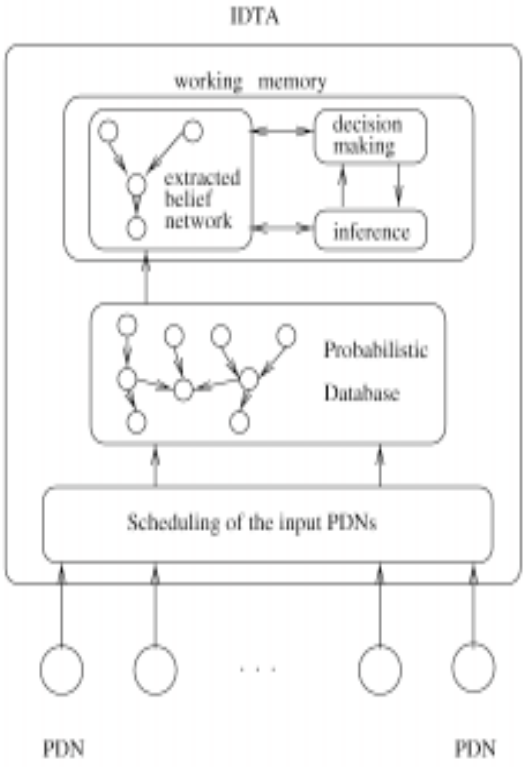


Illustration of an IDTA

Distributed, Intelligent Agent Based Fault Management Architecture

- **Each DDA consists of the following components:**
  - Intelligent Monitoring and Detection Assistant (IMDA). Monitors and analyzes the data and classifies the input data to a set of symptom types.
  - Intelligent Domain Trouble-shooting Assistant (IDTA). Based on the symptoms reported by the IMDA, find the most possible cause and come up with a suggested test sequence
  - Intelligent Communication Assistant (ICA). Helps the DDA to send messages in cases of global problems
- **Each IMDA monitors the network elements (NE) that belong to its network domain**
  - Identifies the specific type of the fault that has occurred
  - IMDA outputs (Problem Definition Nodes (PDNs)), and relevant activation status, each one modeling a certain type of fault.
  - Five activation levels for each PDN to reflect the severity of the relevant network fault: alarm, major, minor, warning, normal

Distributed, Intelligent Agent Based Fault Management Architecture

- **Proposed scheme extracts a sub-belief network, for each PDN based on its activation status, from a database of belief networks, representing the influences of the various causes to the possible network faults**
  - Then the inference and trouble-shooting begins using belief computations and selection of tests using decision theoretic methods
- **Exact inference in an arbitrary belief network is NP-hard,**

  Approximations based on Monte Carlo simulations; importance sampling
- **Fault management system can "learn" belief network representations from data or by capturing the interpretations, inferences and actions of experienced network operators**
- **Maximum a posteriori (MAP) methodology in estimating these parameters**
- **Metropolis samplers and importance sampling algorithms inspired by similar estimation problems in Markov Random Fields over graphs**

Automated Fault and Performance Management

- **Formulated proactive fault management and combined fault and performance management as a dynamic control problem with partial information**
  - Training of belief networks and MAP computations on graphs
  - Progressive learning from small-samples
  - Learning “faulty patterns” via neural networks
  - Correction strategies via reinforcement learning and Q-learning
- **Investigated dynamic routing and CAC coupled to proactive fault and performance management; Fast algorithms**
- **Investigated fast computation of network response surfaces from simulations/analytical approximations**
- **Initiated implementation in G2/Neuronline/OPNET/ HPOpenView/ORACLE environment**